UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Technology
Robotics and Computer Engineering Curriculum

Elmar Abbasov

Improving object detection in adverse weather conditions for Auve Tech's autonomous vehicle

Master's Thesis (30 ECTS)

Supervisors: Prof. Gholamreza Anbarjafari, PhD
Asif Sattar, MSc

Tartu 2022

Acknowledgements

My sincerest gratitude goes to the faculty of the Institute of Technology, whose teachings and trust in me have already led to my dream career for the better half of a decade.
This work wouldn't have been possible without invaluable feedback and patience from my supervisors Prof. Gholamreza Anbarjafari and Asif Sattar.
I'd also like to thank Auve Tech and specifically Ott Männik, for supporting this research and being a joy to work with.
I would be remiss in not mentioning the help I got in the translation of the abstract from my dearest friends Veronika, Tõnu, Kätlin, Janno and Kadi.

Improving object detection in adverse weather conditions for Auve Tech's autonomous vehicle

Abstract:

As autonomous vehicles are becoming more prominent in our lives we want their computing systems to be able to recognize objects with the best accuracy possible, regardless of the weather conditions. In order to achieve better accuracy with machine learning based visual object detection we compare 2 approaches: training an object detection neural network with synthetic rain added images and removing rain from images using a different state-of-the-art neural network before feeding them to an object detection neural network trained with non-rainy images.

In this work, first, we generate custom datasets for training and evaluation as no dataset fulfilling the requirements of this project is publicly available. After that, using various combinations of these datasets for training and evaluation, we arrive at our main findings: We show that while using a specialized neural network for removing adverse weather effects from images results in visually more distinguishable objects in these images, an object detection network doesn't perform better in this scenario than if it's trained with labeled images with adverse weather effects on them. We perform the main experiments with synthetic rain, and then confirm our findings with images taken in real-life rainy situations. The outcome of this research will be used in the computer vision pipeline of Auve Tech autonomous vehicles.

Keywords:
Object detection, YOLO, deraining, autonomous driving, adverse weather conditions

CERCS:
T111 Imaging, image processing
P176 Artificial Intelligence

Objektide tuvastamise parandamine ebasoodsates ilmastikutingimustes Auve Techi autonoomse sõiduki jaoks

Lühikokkuvõte: Autonoomsed sõidukid muutuvad üha suuremaks osaks meie elust. Selle tõttu tahame, et nende süsteemid oleks võimelised tundma ära asju võimalikult täpselt sõltumata ilmast. Saavutamaks täpseimat tulemust masinõppel põhineval visuaalsete objektide tuvastamisel võrdleme me kahte lähenemist: Objektituvastamise treenimist närvivõrkude abil, piltidega millele on eelnevalt töödeldud sünteetiline vihm ja vihma eemaldamist piltidelt, kasutades uusimaid närvivõrke enne pildi lisamist objektituvastamise närvivõrku, mis on treenitud vihmavabade piltidega.

Selles töös genereerime esmalt piltide andmebaasid treenimiseks ja hindamiseks, sest ühtegi vajadusi rahuldavat andmebaasi pole avalikult saadaval. Peale seda kasutame erinevaid kombinatsioone nendest andmebaasidest. Kuigi me näitame, et kasutades spetsiaalseid närvivõrke ebasoodsate ilmastiku efektide eemaldamiseks piltidelt saame tulemuseks visuaalselt rohkem eristatavad objektid neil piltidel siis sellisel viisil objektituvastus ei anna paremaid tulemusi kui vihmase ilmaga piltidega treenitud võrk. Me teeme peamiseid eksperimente sünteetilise vihmaga ja siis kinnitame oma leide päriselust võetud vihmaste olukordade piltidega. Antud uurimuse tulemust kasutatakse Auve Techi autonoomsete sõidukite jaoks.

Võtmesõnad:
Objekti tuvastus, YOLO, vihma-eemaldus, autonoomne liikumine, ebasoodsad ilmastikuolud

CERCS:
T111 Pilditehnika
P176 Tehisintellekt

Contents

1 Introduction

Autonomous vehicles are rapidly shaping up to become an important part of our lives. The main aim of such vehicles is to improve vehicle navigation, save traveling time and reduce road fatalities. The latest is the most important as crash injuries are now estimated to be the 8th leading cause of death globally for all age groups and the leading cause of death for children and young people between 5 to 29 years of age [1]. Other benefits of such systems include significantly reducing human labor involved in the transport of humans and goods.

One startup working on self-driving technology is Auve Tech, which is developing an autonomous transportation system based on the concept of a shuttle bus. This vehicle is aimed at enhancing last-mile transportation by offering alternative means of transport in closed areas and mixed traffic environments and is powered both by electricity and hydrogen.

In order to facilitate this transportation system, we, at the software team of Auve Tech, have been implementing state-of-the-art software solutions to various challenges faced by today's autonomous systems. One of such challenges is object detection and classification, which is necessary for the shuttle to comprehend the traffic around and react to it accordingly. Recent advancements in the development of the deep learning class of machine learning algorithms combined with increased computational resources available for edge devices and the availability of huge image/video based datasets of target object classes have enabled relatively fast and accurate detection and classification of these objects.

Considering these, at Auve Tech, we utilize state-of-the-art neural networks for creating a computer vision-based object detection and classification pipeline. While Auve Tech's shuttle buses are already deployed around the world and serving travelers autonomously, we keep improving the safety and the abilities of its autonomous system.

1.1 Motivation

While most of the available state-of-the-art solutions for computer vision-based object detection and classification are generally fast and powerful, they lose precision when faced with adverse effects of the weather as they are not trained to deal with such conditions. This is an expected default behavior as the objects in the view of the camera are distorted from the perspective of the camera and may be occluded by the rain, snow, or other similar occlusions. This makes the off-the-shelf trained object detection models insufficient for application and requires special treatment for detecting objects in these adverse conditions.

An apparent solution would be training the network with a big amount of data in these domains, as deep neural networks are, as pointed out earlier, data-hungry and may require enormous amounts of data to train. However, very few such datasets exist

and the lack of datasets captured in diverse domains like rain, snow, and other weather conditions represents one of the most challenging aspects of achieving a viable level of training for autonomous vehicles.

1.2 Objectives

In the scope of this work, we create a computer vision based object detection and classification pipeline for Auve Tech shuttles that will perform better in adverse weather conditions than currently available solutions. Specifically, We will prepare a custom data set consisting of images from Argoverse data set [2] with varying levels of synthetic rain added. Further, We will review and choose a state-of-the-art deraining solution. Then, we will evaluate and compare various recent computer vision based object detection and classification implementations. Thus, We will introduce the best pipeline in terms of accuracy considering the real-world computing resources available on the shuttles of Auve Tech.

We will be comparing two different pipelines to verify our hypothesis of object detection in images with adverse weather conditions. Figure 1 shows these two pipelines, where the only difference between the two pipelines is the direct utilization of rainy images for classifying images in the first pipeline. Whereas, the second pipeline first restores rainy images, so that the effect of rain is mitigated and then a neural network will be trained for object detection using the resultant images.

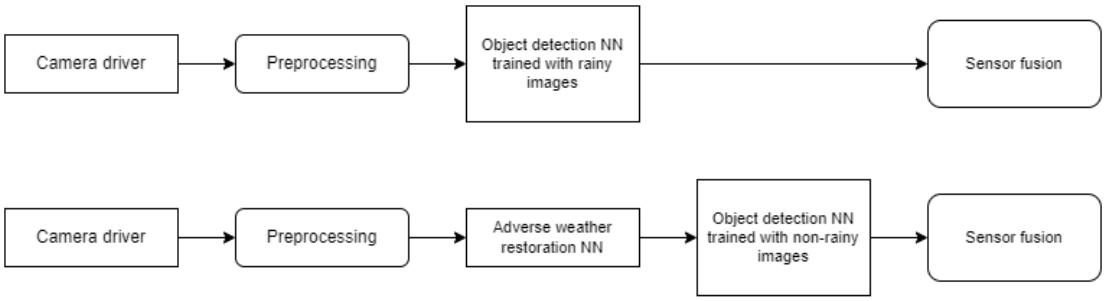


Figure 1. Two pipelines we are comparing our hypothesis with.

2 Background

2.1 Auve Tech

Auve Tech [3] is a technology startup that specializes in the development and manufacturing of autonomous transportation systems. It started as a spin-off of a research and development project from IseAuto research group of Tallin University of Technology. The company has already been providing commuters with autonomous rides in 4 different countries, and its engineers are working hard on scaling up the technology further. In addition to its operations, the project has also been a subject of many research papers [4–9] and theses [10–18].

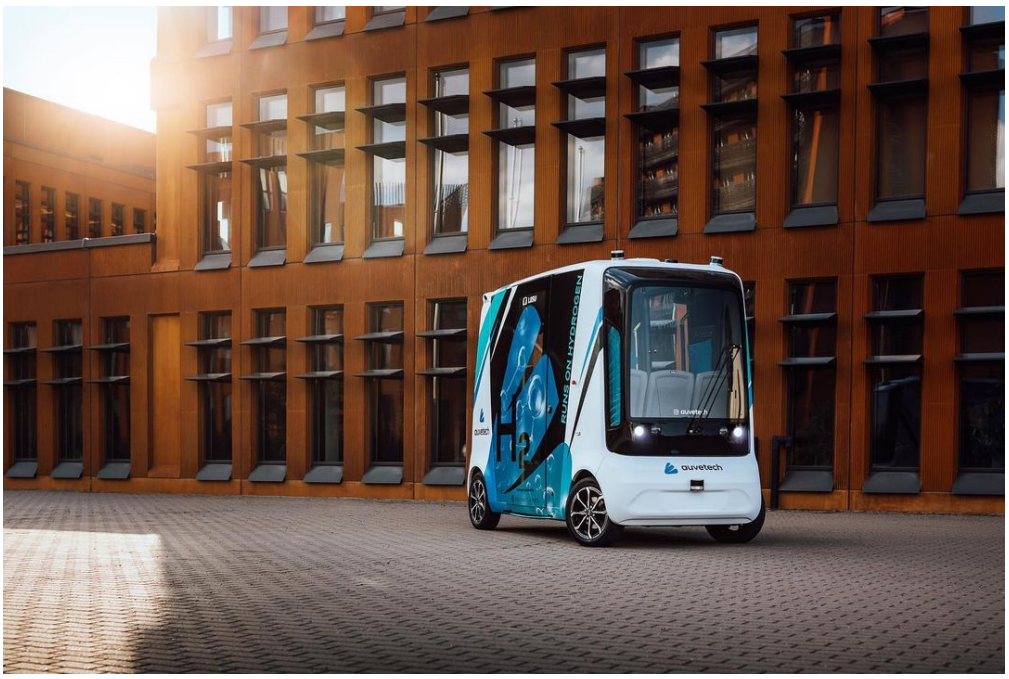


Figure 2. Auve Tech's shuttle bus powered by hydrogen

Auve Tech is focused on providing smart and sustainable last-mile transportation to its commuters. Therefore, in 2021, it introduced a low-temperature cell-powered autonomous shuttle bus, which allowed the vehicle to produce energy from hydrogen right inside the shuttle. The development of this shuttle bus technology was conducted in collaboration with researchers from University of Tartu [19]. Figure 2 shows the trademark shuttle bus vehicle designed by Auve Tech for its commuters.

One of the primary goals of the company is to continuously increase the scope of the conditions this shuttle bus is able to withstand and safely navigate autonomously.

Reducing the effects of adverse weather conditions on the sensor suite is one of the ways to reach that goal. The same is the purpose of this research work, so it is aligned with the primary goals of the company.

2.1.1 Hardware setup

The sensor setup for the shuttle bus is shown in Figure 3, with some of the sensors related to object detection clearly marked. Some of them are describe below:

- 3 x Long range Lidar sensors for wide range pointcloud data collection.
- 4 x FLIR Blackfly S cameras with fisheye lenses, 1 per each side of the vehicle capturing images at the resolution of 1616x1240 and up to 30 FPS
- Inertial Measurement Unit
- Global Navigation Satellite Systems (GNSS) receiver and antenna

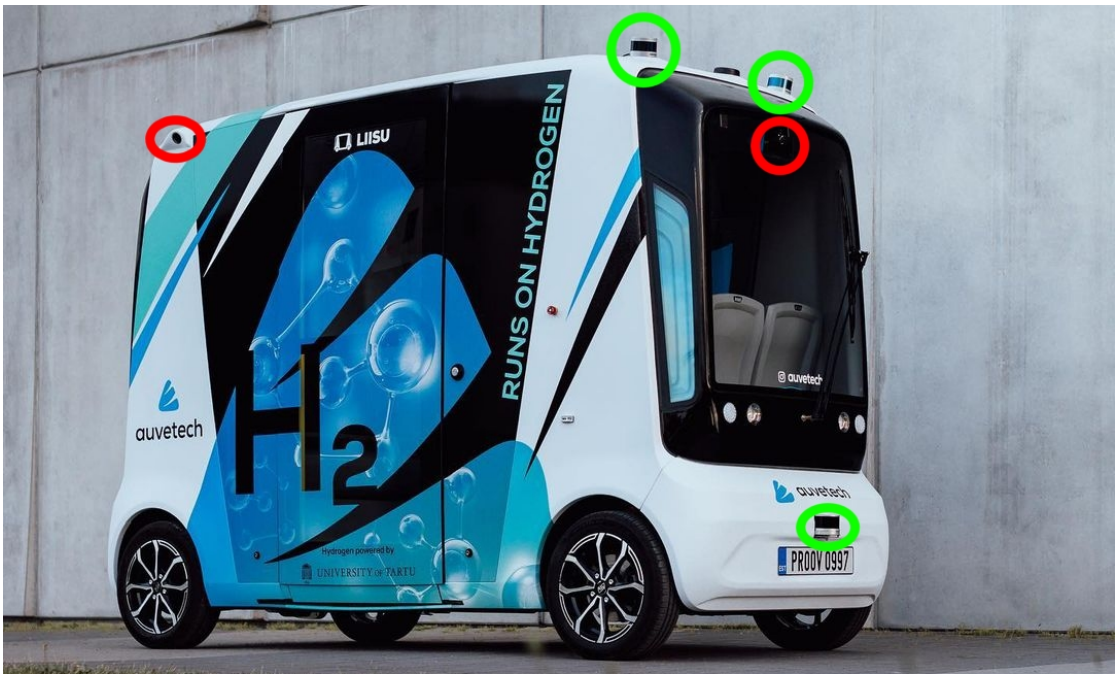


Figure 3. Sensors on Auve Tech shuttle relevant to object recognition: Camera positions are marked with a red circle, lidar positions are marked with green circles [3]

.

We are not able to share the exact computational setup of the vehicle as the information is restricted, but the main computer's performance is in par with a high-end desktop

PC with the main GPU for inferencing being on the level of a desktop Nvidia RTX 2070 card.

2.2 Autoware

Software enabling autonomous driving capabilities of Auve Tech's shuttle buses use a heavily modified version of Autoware [20] as its base, which is an open-source software stack providing functionalities required for autonomous driving in a modular architecture. Autoware itself is built upon Robot Operating System (ROS) [21], which is an open-source meta-operating system for robots. ROS provides the services one would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly used functionality, message-passing between processes and package management.

ROS also provides tools and libraries for obtaining, building, writing and running code across multiple computers. The ROS runtime graph is a peer-to-peer network of processes (called ROS nodes) that are loosely coupled using the ROS communication infrastructure. ROS implements several different styles of communication, including synchronous communication over services, asynchronous streaming of data over topics, and storage of data on a Parameter Server [22].

Autoware provides a helping set of self-driving modules composed of sensing, computing, and actuation capabilities. Capabilities, although rough around the edges and not totally complete, include Localization, Mapping, Object Detection & Tracking, Traffic Light Recognition, Mission & Motion Planning, Trajectory Generation, Lane Detection & Selection, Vehicle Control, Sensor Fusion etc. High level module architecture of the design implemented with Autoware is shown in Figure 4.

The part of the stack we're dealing with in this work is related to the object recognition pipeline, shown in Figure 5. More specifically, we're looking into the image based object detection part of object detection pipeline. The detected objects from this stage of the pipeline is then fused with detection results based on data from lidar sensors' pointcould data. This fused object data then passes through shape estimation and moves on to Multi Object Tracking algorithms in order to analyze the objects' speed and acceleration, as well as giving them unique identification numbers. As we by this point know the type of the object (Whether it's a car, a pedestrian or some other object) and its speed, by adding the knowledge we know about the current location and the structure of the road, we can make assumptions about the future behavior of the fellow road participants.

It can be seen that many steps in the pipeline depend on the initial step of object detection. Hence, in order to facilitate high accuracy in the later parts of the pipeline, it's imperative to get as high quality of the initial detection results as possible.

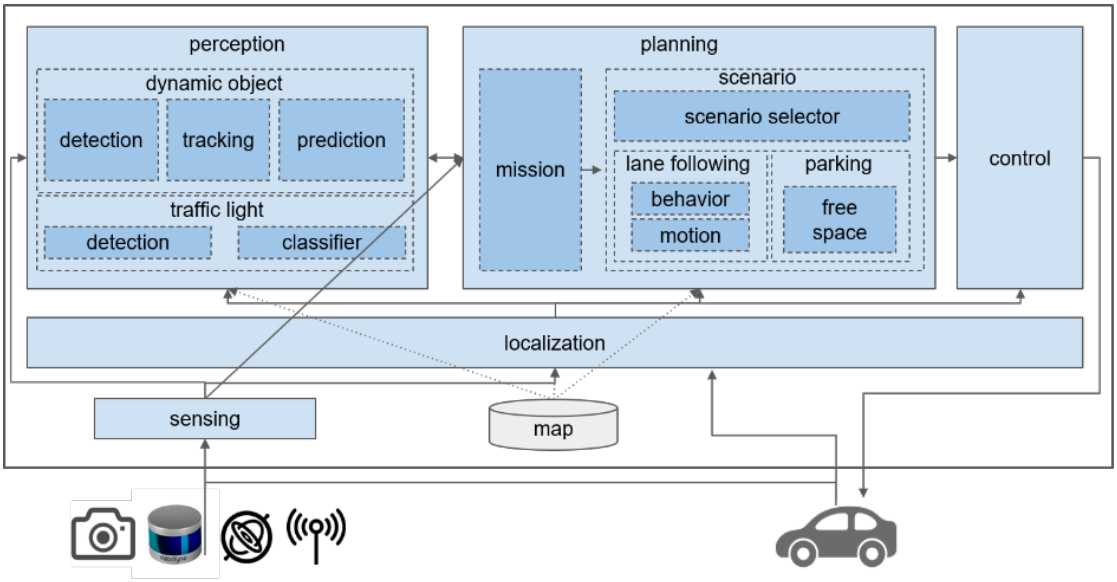


Figure 4. High level architecture of Autoware [23]

2.3 Visual object detection and classification

As visualized in Figure 5, detections and classifications from camera images in the initial detection phase inform the later stages of sensing pipeline of the autonomy system and therefore it's crucial to get detections as accurate as possible in order to ensure our Multi Object Tracking and Map based Prediction algorithms don't get fed confusing information and make wrong decisions. Nowadays, across many companies and research institutions this complex task is reserved for deep learning based neural networks as they have shown good results in detection of objects like pedestrians, vehicles, traffic lights and traffic signs [24–29]. These networks are usually trained with a large amount of data and predict location of certain objects in the given image.

One branch of such neural networks like R-CNN [24] suggested a two-stage approach: The first stage is for determining Regions Of Interest, areas in the image that are likely to hold objects of interest; The second stage is where class probabilities are determined for the corresponding Regions of Interest. This branch gets later extended with Fast R-CNN [25], which is 9 times faster and manages to achieve significantly better precision score compared to R-CNN. A year later in 2016, after it was recognized that this multiple stage detection added computational cost, Ren et al. proposed Faster R-CNN [30], which tries to alleviate the performance cost by introducing Region Proposal Network(RPN) that shares full-image convolutional features with the detection network. Architecture of Faster-CNN is illustrated in Figure 6

A different approach, however, started to give great results around the same time as

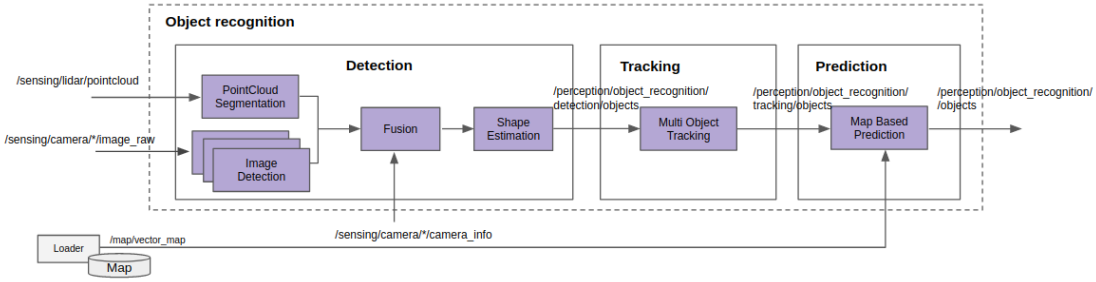


Figure 5. Architecture of the object recognition pipeline. Each purple rectangles represents a ROS node, and the arrows represent the publisher/subscriber relationship between them, and the accompanying text represents the relevant topic name for the communication [23]

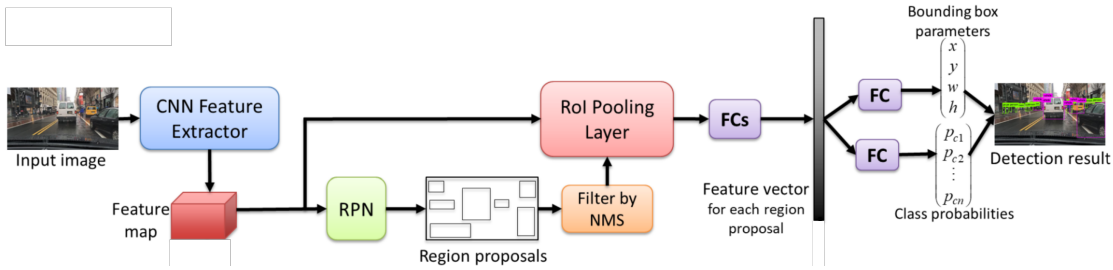


Figure 6. Architecture of Faster R-CNN [31]

Fast R-CNN was presented. Redmon et al. proposed YOLO [29], which took a one stage approach to object detection and managed to reduce computational complexity enough to make decent object detection feasible in real-time. Comparison of real-time performance of R-CNN derivatives and YOLO are mentioned in Table 1.

2.3.1 YOLO (You Only Look Once)

As this work heavily relies on descendants of YOLO [29], it makes sense to look into the original paper and some of the state of the art methods inspired by it that have become go-to solutions for real-time object detection ever since.

In 2015, Redmon et al. changed the way, the image-based object detection is achieved, with the paper "You Only Look Once: Unified, Real-Time Object Detection" [29]. For the first time, object identification has been framed by them as a regression task to detect spatially separated bounding boxes and their associated class probabilities, instead of using classifiers for detection. For this purpose, separate neural networks was used to predict the bounding boxes and class probabilities from a single image. The class probabilities are only predicted for a bounding box if an object of interest is actually detected inside the bounding box.

As a very fast, customizable and generalizable solution, this method has inspired many descendants that are being used by many real-time state of the art solutions, including in autonomous driving pipelines. As many new flavors of YOLO are released every year and we are focused on state of the art, for this work we mainly focus on recent releases.

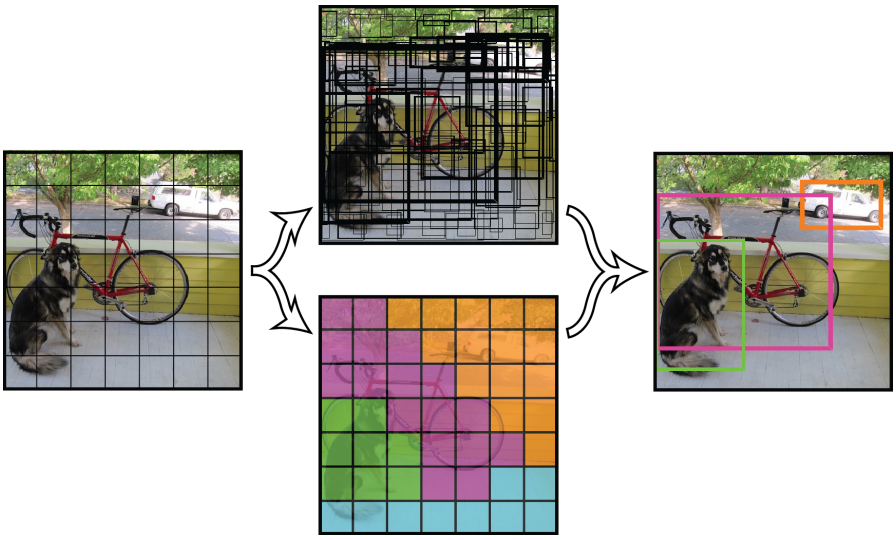


Figure 7. Example from the original paper [29].

2.3.2 YOLOv1

In the original YOLO, a feature extraction module comprised 24 convolution layers. After this, 2 fully-connected (FC) layers are used for object classificaiont as well as detecting bounding boxes (as regression coordinates of bounding boxes). A 7x7x30 tensor is produced as an output of the network. Figure 8 shows the archictecture block diagram for the original YOLO model.

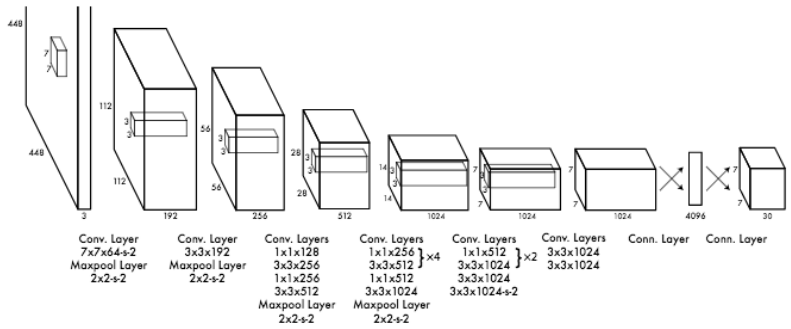


Figure 8. Architecture of the original YOLO [29].

Table 1. YOLO real-time performance in comparison with R-CNN derivatives based on training with PASCAL VOC dataset [29]

Network	**mAP**	**FPS**
R-CNN [24]	53.5	6
Fast R-CNN [25]	70	0.5
Faster R-CNN VGG-16 [30]	**73.7**	7
Faster R-CNN ZF [30]	62.1	18
YOLO [29]	63.4	**45**

2.3.3 YOLOv3

'YOLOv3: An Incremental Improvement' [32] is the last paper from the original researchers of YOLO, Redmon et al. The third version has seen big improvements compared to the original in terms of speed, precision and the specificity of classes. YOLOv3 now uses Darknet-53, a 53 convolutional layer feature extractor as its backbone, which is also made by the YOLO creators Joseph Redmon and Ali Farhadi. Figure 9 shows the archictecture block diagram for the original YOLO model. YOLO-v3 demonstrated much superior performance compared to its competitors at the time of its development in terms of mean Average Precision (mAP). Figure 10 shows the comparison of YOLOv3 with a couple of other model RetinaNet-50 and RetinaNet-101 [26].

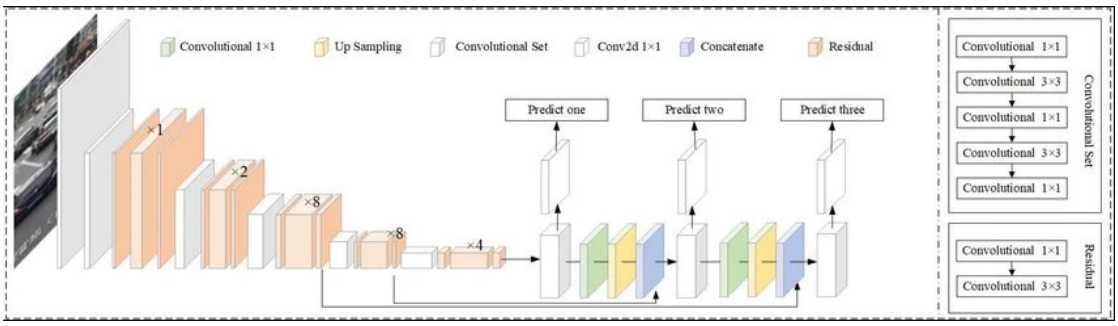


Figure 9. Architecture of YOLOv3 [32].

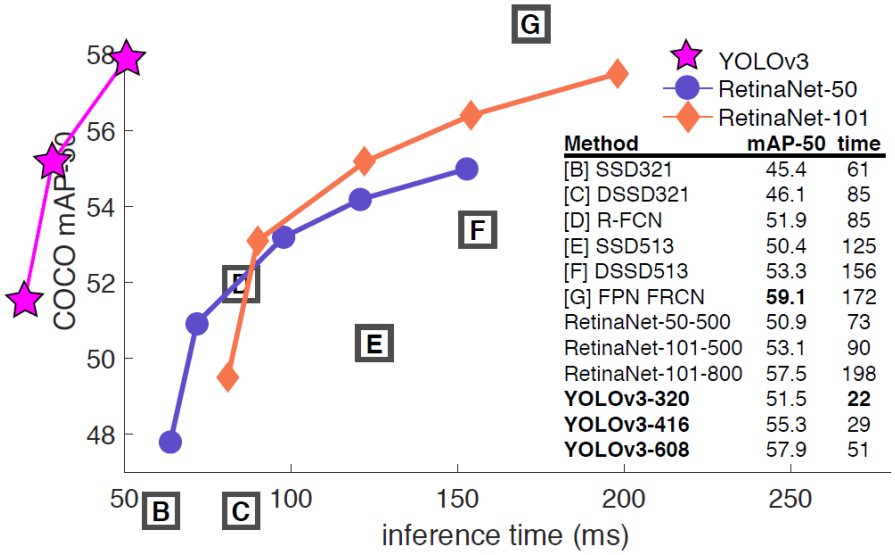


Figure 10. YOLOv3 performance [32].

2.3.4 YOLOv5

While not by original YOLO researchers and not accompanying a paper, YOLOv5 [33] is a highly popular YOLO descendant that was originally released in 2020. YOLOv5 has gathered a big community of contributors around it, which is why its new versions keep getting released with better performance and new integrations. The first large contribution of YOLOv5 was that it translated a very efficient Darknet framework to the corresponding implmentation in the PyTorch framework. C is the primary programming language used in this Darknet framework, which provides fine-grained control over the network's activities. It may be a benefit to research to have control over lower level languages, but it can also slow down the research process since one must develop custom gradient computations for each new addition.

YOLOv5 also implements data augmentation processes that happen before training in order to improve the Average Precision considerably. One of these methods is called the mosaic data augmentation, which was a creative way to combine multiple (actually 4) training images stacked on top or side of each other. Due to this, the model was able to learn identifying the objects at normal as well as smaller scales. This had added benefit of reducing the training time as the requirement of large mini-batch sizes was now alleviated.

We feel confident choosing YOLOv5 for the object detection and classification part of our pipeline as it offers one of the best trade-off levels between accuracy and speed available, and is also similar in use to YOLOv2 [34], the solution chosen for this project

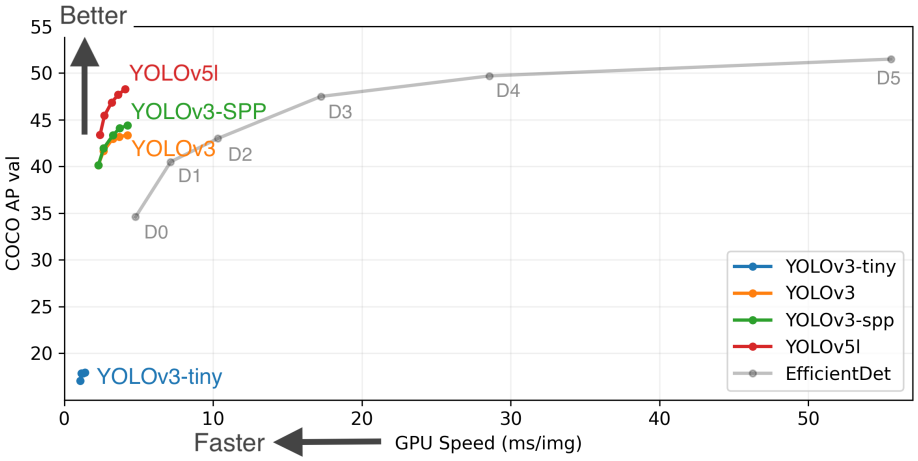


Figure 11. YOLOv5 performance [33].

during a previous research [17]. Figure 11 shows the comparison of YOLOv5 with a few variants of YOLO-v3 and EfficientNet [35] in terms of mAP on the popular COCO data set [36].

2.4 Deraining, Desnowing and Defogging

In order to assist vehicles in safely navigating through traffic, advanced autonomous systems depend on visual data to identify and localise various objects, including people, traffic signs, and other vehicles, for the purpose of providing assistance. However, in challenging weather scenarios such as rain, snow or fog, the performance of object detection methods suffer significantly [31]. One of the ways we can combat this situation is using state of the art neural networks to derain (remove the effect of rain from images), desnow and/or defog the image before feeding it to an object detection and classification algorithm like YOLOv5.

While discussing rain, snow or fog in the context of autonomous driving, the differences between them for vision algorithms must be understood: The vast majority of traditional algorithms that are used in outdoor vision systems begin with the presumption that the picture intensities are proportional to the brightness of the scene. Nevertheless, dynamic weather circumstances such as rain or snow create dramatic intensity variations in images, which proves that this assumption is incorrect. [37]. Raindrops on glass surfaces in front of the camera also cause extra disturbance because they cause blurry edges behind themselves and cause the refraction of light [38].

Potential approaches to solve this problem vary and there are some techniques reviewed is some review studies. For example, Mazin Hnewa and Hayder Radha [31]

reviewed and tested some approaches of deraining. But, in the current work, we will review some of the latest research work that utilizes the vision transformers; a method that's recently gaining attraction in computer vision applications for its outstanding results. Figure 13 shows the Transweather model architecture, and Figure 12 demonstrates an example image-pair showing the images before and after the application of Transweather. It can be seen that the output image after the application of Transweather is restored successfully and the effects of rain pixels are largely removed.

2.4.1 Transweather

Figure 12. Example input and output from Transweather [39]

Transweather [39] is an effective solution based on the idea of transformer for the removal of any unfavourable weather conditions. It makes use of a single encoder-decoder network for the restoration process, and it makes use of learnable weather type queries inside the decoder. These queries allow it to learn the kind of weather deterioration that occurred, and then it utilises this knowledge for the weather removal process.

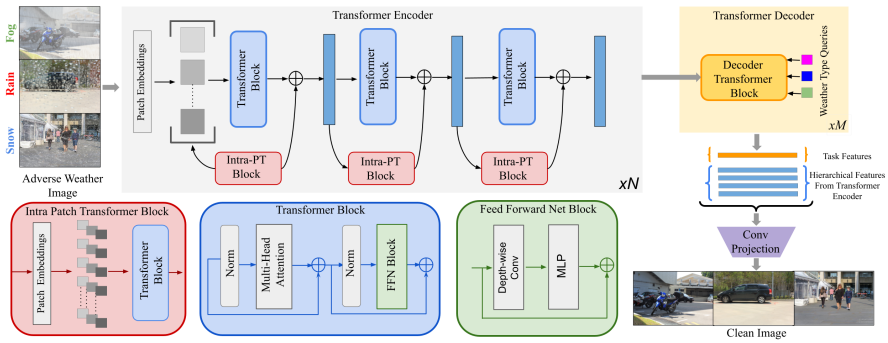


Figure 13. Transweather architecture [39].

3 Datasets

Dataset used for training and validating the neural network is one of the most important parts of visual object detection and classification solution. We are, however, quite limited in datasets we are able to use for autonomous driving in adverse weather conditions. The go-to datasets popular among Computer Vision researchers for training Neural Networks like COCO [36], KITTI [40], Argoverse [2] or CityScapes [41] have no labels regarding the weather conditions in the pictures. There's only BDD100K dataset [42] that we have managed to find that fulfills the requirements for such work. Even then, as pointed out in [31], the quality is of labeling and diversity in the dataset is sub-optimal. Another approach to demonstrate the impact of rain on object detection methods is to render synthetic rain on top of images taken in clear weather conditions.

3.1 Argoverse

Argoverse 1.1 is a self-driving dataset collected by Argo AI in order to assist autonomous driving research. It contains data from 2 lidar sensors, 7 ring cameras and 2 front-facing stereo cameras as illustrated in Figure 14.

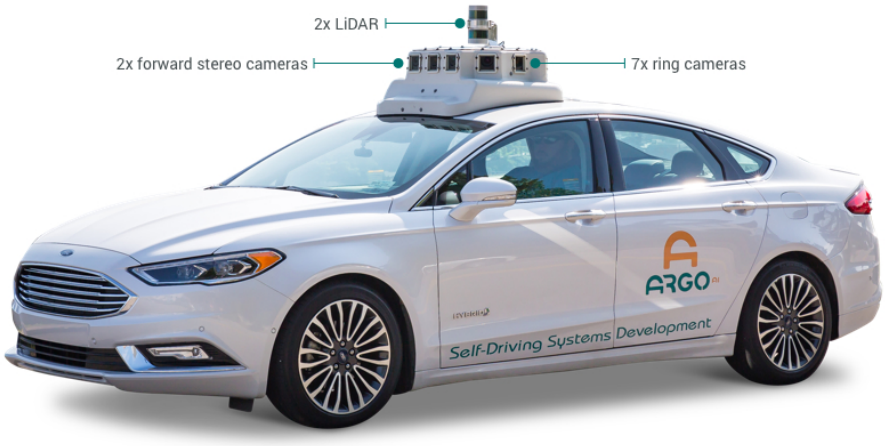


Figure 14. Argoverse data collection vehicle [2].

For this work we take output from one of the front facing ring cameras, which includes 39384 labeled images in the training and 15062 labeled images in the validation subset of the dataset.

3.1.1 Augmentations on Argoverse dataset

We added 3 different levels of synthetic rain to our subset of Argoverse dataset before using it for training and validation of the Neural Network. For this purpose, we using the image augmentation library imgaug [43]. The following proportions were implemented for augmentation: 40% Light rain, 40% Medium rain and 20% Heavy rain. Figure 15 shows the examples of these different levels of rain occlusion on a sample image. It can be observed from the figure that the image becomes more challenging with each level of rain added.

Figure 15. From top to bottom: Light, Medium and Heavy synthetic rain

3.2 BDD100K

BDD100K [42] (Berkeley DeepDrive) is a very large and diverse open driving video and photo dataset for computer vision research. Dataset consists of 100,000 videos, hence the 100K in the name. Each video is about 40 seconds long, 720p, and 30 fps. The videos also come with GPS/IMU information recorded by cell-phones to show rough driving trajectories. The videos were collected from diverse locations in the United States, and the locations are marked in Figure 16.

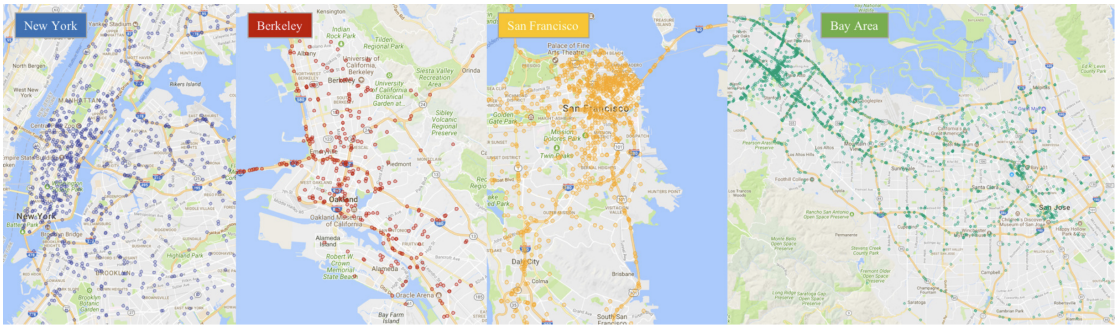


Figure 16. Geographical distribution of data sources from BDD100K. Each dot represents the starting location of every video clip [42]

The subset of BDD100K we'll be using are the annotated images that are a part of the dataset: A keyframe at the 10th second from each video is selected and sampled, with annotations provided for these samples. They are labeled at several levels: image tagging, road object bounding boxes, drivable areas, lane markings, and full-frame instance segmentation. These annotations also include the weather conditions in the image, which is very important for our work. An overview of the annotations available is demonstrated in Figure 17.

3.2.1 Limitations with the BDD100K

It'd be great to be able to use BDD100K for training part of our research, as it technically fits all our requirements and contains real world rain data. However, there is an issue with the dataset as it's published: As mentioned in [31], weather condition labels on BDD100K are not all accurate. Some of the images marked as rainy only has wet asphalt, some others are taken plainly in clear weather although marked rainy. If we sort through all the images marked to be rainy, we find only 1596 images. For this reason, we'll be using this subset of BDD100K only as a dataset to validate our strategies.

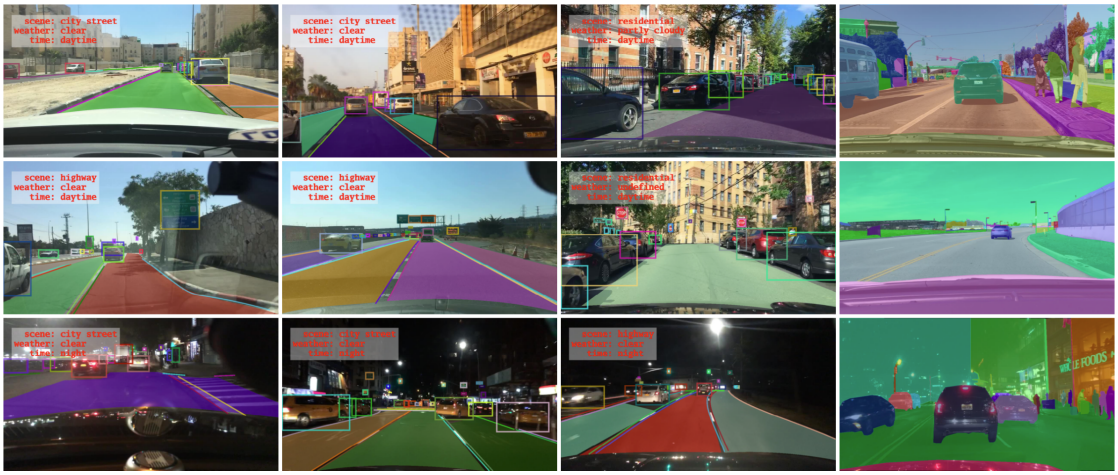


Figure 17. Overview of annotations available for key frames of videos from BDD100K [42]

3.3 Processing of images with Transweather

We used Transweather [39] on our custom version of Argoverse validation dataset with synthetic rain added, and the results were quite good (see Figure 18). Figure shows different amounts of rain added synthetically to an image and the corresponding output image, in each case, after applying the Transweather model. The ground truth image is also given in the bottom row for fair comparison.

We also obtained quantitative performance measures for the results produced by the Transweather model. For this purpose, we utilized Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Metric (SSIM), which are standard measures of image quality based on comparison between a pair of images (the ground truth and the query/resultant image). We got PSNR of 29.36 dB and SSIM of 0.9376 for our dataset. We get these values in comparison with the ground truth images, and they're quite good performance ratios as explained in the Section 4.1.

3.3.1 Performance of Transweather

One key issue we are now able to determine on using Transweather in our pipeline is the computational complexity this adds. According to the experiments in [39], inference time of Transweather is the shortest compared to alternative methods at 0.14 seconds while delivering images with less noise. Our experiments confirm this also, as we managed to process 7 frames per second when the computing hardware isn't loaded with any other task. While this is the best performing option of this method, usability of this in real-time while sharing computing resources with other important tasks would

require a tremendous amount of optimizations.

Figure 18. From top to bottom: Input, Output, Ground Truth

3.4 Datasets defined for use in experiments

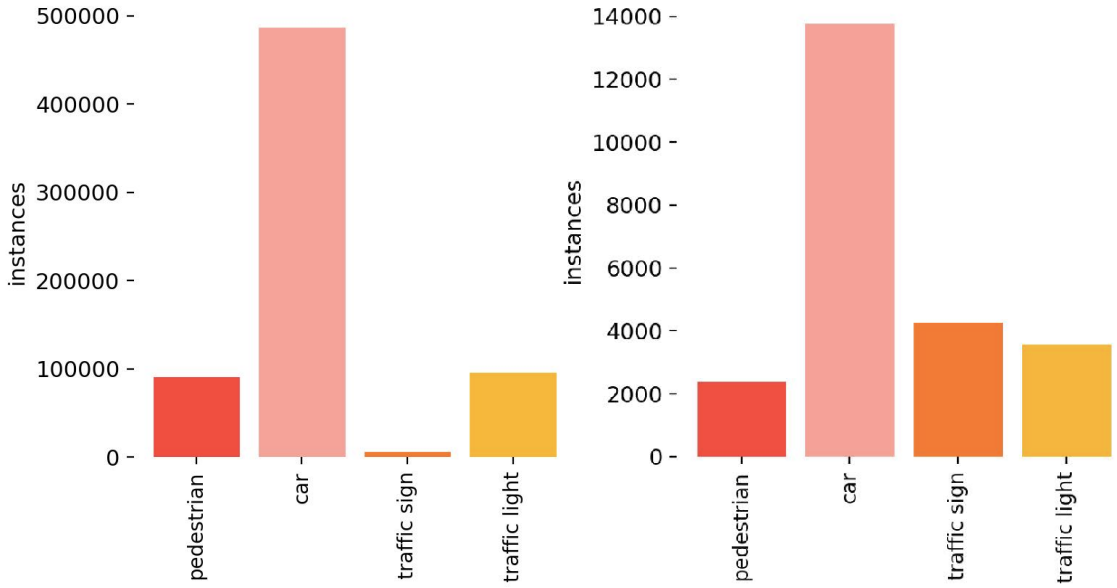
We will go over the details of the datasets we used for experiments with the results mentioned on the Table 2.

3.4.1 Clean

For Clean dataset, we will use images from Argoverse mentioned in the Section 3.1 without any modifications. The number of instances appearing in the dataset are shown in Figure 19a.

3.4.2 Synthetic Rain

Synthetic Rain dataset uses the same images as the Clean dataset, with synthetic rain added on top of them like described in Section 3.1.1 and illustrated in Figure 15.



(a) Clean/Synthetic (b) Real

Figure 19. Number of class instances in the datasets.

3.4.3 Real Rain

Real Rain is the subset of BDD100K we talk about in the Section 3.2.1. It contains the images from BDD100K manually selected and confirmed to have real rain drops in the image. Because of its small size, we use it only for validation and not training. The number of instances appearing in the dataset are shown in Figure 19b and examples can be seen in the top row of Figure 20.

3.4.4 Synthetic Derained

Synthetic Derained has the images from the Synthetic Rain dataset that we further modified by using Transweather [39] to remove the rain streaks from the images. Examples of the results can be seen in the middle row of Figure 18. This will be used as one of the validation datasets in our experiments.

3.4.5 Real Derained

Real Derained dataset has the images from our Real Rain dataset that has gone through deraining with Transweather [39]. Examples can be seen in the bottom row of Figure 20. Also in the same figure, the modifications Transweather makes in the images can be seen.

Figure 20. Images from BDD100K based datasets. Top: Real Rain; Bottom: Real Derained

4 Experiments

In this work, we have designed experiments for determining the best state of the art pipeline for visual object detection for the Auve Tech's autonomous shuttle busses. We have run various experiments wherein different augmentations are applied to different parts of the pipeline in order to determine the best combination. We have also considered the accuracy of object detection and its performance implications, as we'll be running the pipeline in a real-world environment with limited computational resources, alongside other resource-intensive software components.

The 2 main experiments for this research are given as following:

- Removing rain from images using a different neural network before feeding them into the object detection neural network;
- Training an object detection neural network with synthetic rain added images;

The results of these experiments are compared in order to establish our modified object recognition pipeline for rainy weather conditions. In order to reach that point, we have to perform a few other interesting experiments for determining the best fitting architectures for object-detection/deraining neural networks on different training datasets.

This is detailed in different sub-sections of this section, but before that we need to define a few performance metrics used in this work, which are defined in Section 4.1 below.

4.1 Evaluation Methods

4.1.1 Peak Signal to Noise Ratio (PSNR)

The peak signal-to-noise ratio (PSNR) [44] is the ratio of a signal's greatest strength to the power of corrupting noise that influences the quality of its representation. The PSNR value is related to Mean Squared Error, which is computed as follows.

$$MSE = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}(x_{ij} - y_{ij})^2 \tag{1}$$

Where, x_{ij} and y_{ij} represent the pixels values in i_{th} row and j_{th} column of the images x and y, respectively, and m, n represent the number of rows and columns in the images, respectively.

Based on MSE, the PSNR between 2 images is calculated as given in the following mathematical expression.

$$PSNR = 10.log_{10}\frac{MAX_I^2}{MSE} \tag{2}$$

Where, MAX_I represents the maximum possible pixel value in the images, and MSE represents the mean squared error computed using Equation 2

4.1.2 Structural Similarity Index Measure (SSIM)

The SSIM [44] is a perception-based model that examines the deterioration of an image as a perceived shift in its structural information. This is calculated between a query image and a reference image by using the average, variance and covariance values of pixels of both the images. Mathematically, it is expressed as:

$$SSIM = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{3}$$

Where, μ_x^2 and μ_y^2 represent the average pixels' value of images x and y, respectively. Similarly, σ_x^2 and , σ_y^2 represent the images' variances. Similarly, σ_{xy} represents the images co-variance and c_1 and c_2 represent 2 constants which depend on images dynamic range.

4.1.3 Mean Average Precision (mAP)

The mean average precision (mAP) measure is used in object detection tasks to measure how well a model is performing object detection for all the classes of interest. This measure is related to the concepts of Intersection over Union (IoU), which measures the amount of intersection of a predicted bounding box with that of the ground truth. Based on the IoU threshold, different true positives (TP) and false positives (FP) can be generated. So, varying the IoU (between 0-1) can lead to a set of values, which can be used to calculate different precision/recall values for a classification model. These precision/recall values can then be used to draw a curve, the area under which measures the Average Precision (AP) of the classification model. Whereas, mean Average Precision (mAP) is the average of the AP measures over all classes of interest. Thus, the mAP measure can be expressed mathematically as follows.

$$mAP = \frac{1}{n}\sum_{k=1}^{n}(AP_k) \tag{4}$$

Here, AP_k is the average precision of k_{th} class computed using the method described above, and n is the number of classes of interest.

4.1.4 Confusion matrix

Confusion Matrix is an NxN square matrix used to calculate different classification measures, where N-1 is the number of classes. For the instances below, we are showing confusion matrices for 4 classes that we are evaluating our networks on. The values you see in the cross areas show the normalized number of predicted values crossing Ground Trues values. False Negative instances where objects are not recognized are represented on the bottom row, and the instances of False Positives where objects are predicted where it doesn't exist are shown on the right column.

4.1.5 Precision/Recall curve

Precision-Recall is a valuable measure of prediction success when the class instances are imbalanced. In information retrieval, precision is a measure of result relevancy, while recall is a measure of how many genuinely relevant results are returned. The precision-recall curve shows the tradeoff between precision and recall for different thresholds. A high area under the curve represents both high recall and high precision, where high precision relates to a low false-positive rate, and high recall relates to a low false-negative rate. High scores for both show that the classifier has both a high precision and high recall values.

4.2 Clean trained YOLOv5 evaluated on Clean

The first step is to evaluate the YOLOv5m 6.1 with original Argoverse dataset's validation set. The mAP for all classes at IoU=0.50 is 0.581. The confusion matrix and the Precision/Recall curve are shown in Figure 21 and 22, respectively.

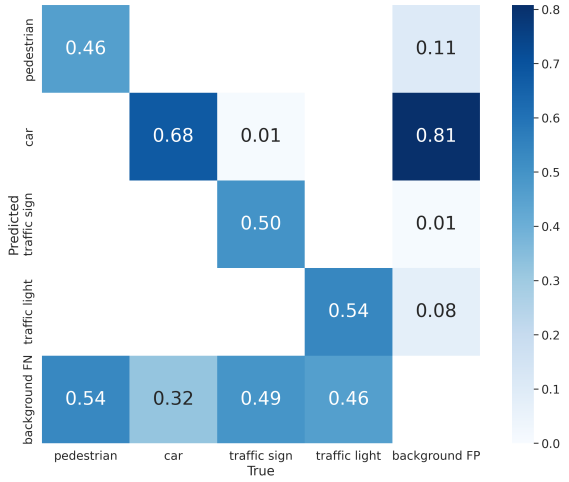


Figure 21. Confusion matrix

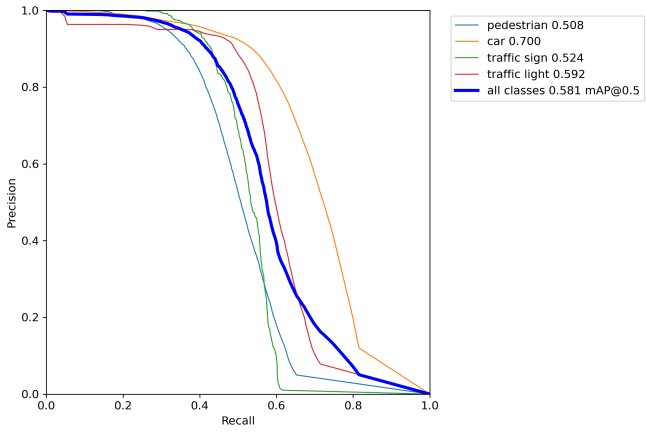


Figure 22. Precision/Recall curve

4.3 Clean trained YOLOv5 evaluated on Synthetic Rain

Then we ran the same test using our custom Argoverse validation dataset with added synthetic rain. The mAP for all classes at IoU=0.50 is 0.510, as expected lower than reference images. The confusion matrix and the Precision/Recall curve are shown in Figure 23 and 24, respectively.

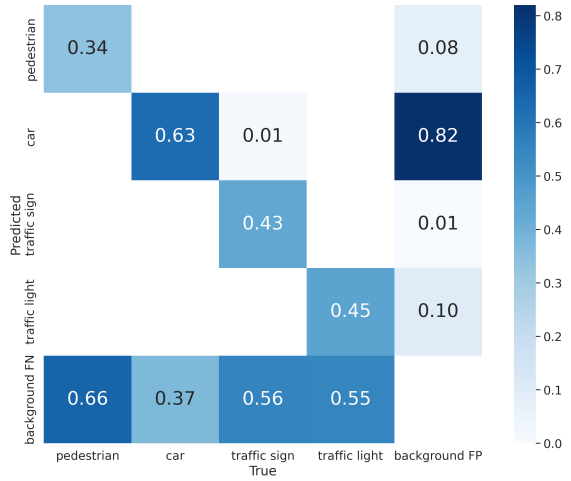


Figure 23. Confusion matrix

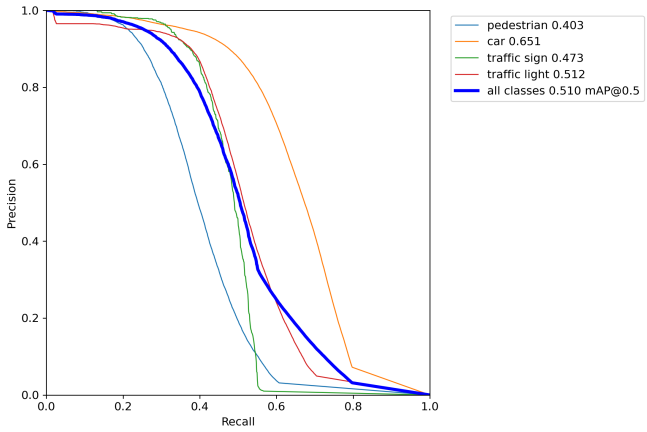


Figure 24. Precision/Recall curve

4.4 Clean trained YOLOv5 evaluated on Synthetic Derained

For the next step we use state of the art deraining solution before feeding the image to YOLOv5 for object recognition. The mAP for all classes at IoU=0.50 is 0.519, which is an inbetween the mean average precision value between the reference images and images the ones with added synthetic rain. The confusion matrix and the Precision/Recall curve are shown in Figure 25 and 26, respectively.

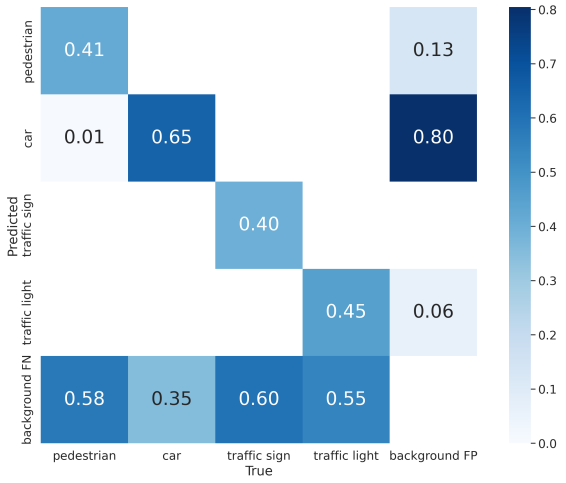


Figure 25. Confusion matrix

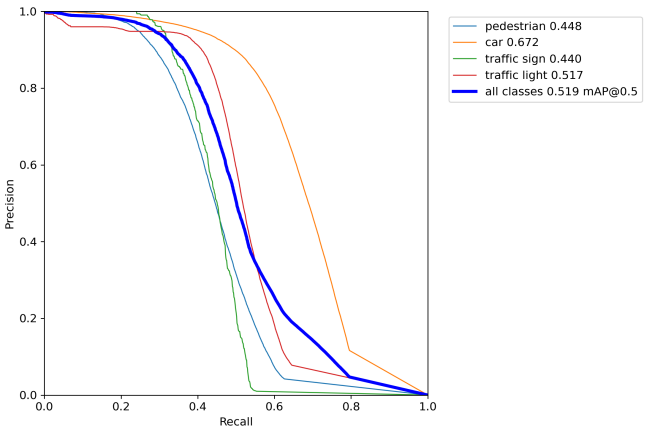


Figure 26. Precision/Recall curve

4.5 Synthetic trained YOLOv5 evaluated on Synthetic Rain

This is where the main comparison point of the research is. While the tests before were done with YOLOv5 trained with non-rainy images, here we test a YOLOv5 that's trained with the training set of our custom synthetic rain added Argoverse dataset. mAP for all classes at IoU=0.50 is 0.555, which happens to be better performing than feeding images through a deraining network before feeding it to YOLOv5. The confusion matrix and the Precision/Recall curve are shown in Figure 27 and 28, respectively.

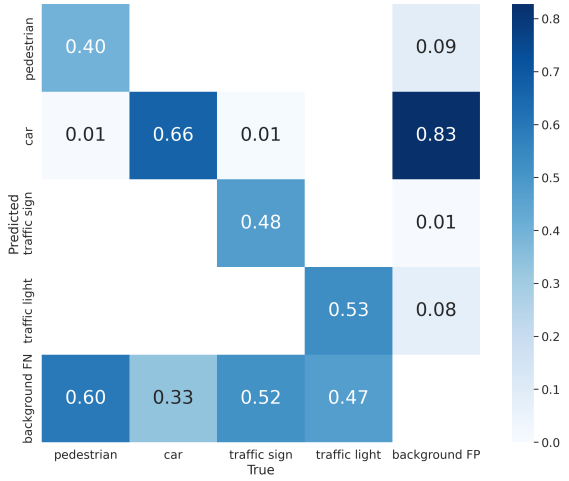


Figure 27. Confusion matrix

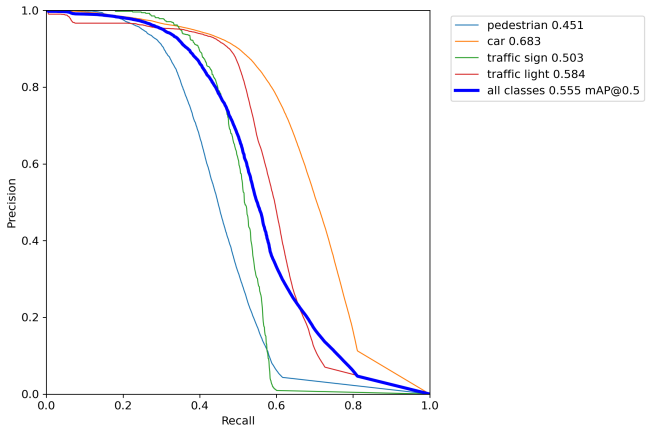


Figure 28. Precision/Recall curve

4.6 Synthetic Rain trained YOLOv5 evaluated on Clean

If we're to use rainy trained YOLOv5, we might have to use it for non-rainy weather also. That's what we test next. mAP for all classes at IoU=0.50 is 0.566. The confusion matrix and the Precision/Recall curve are shown in Figure 29 and 30, respectively.

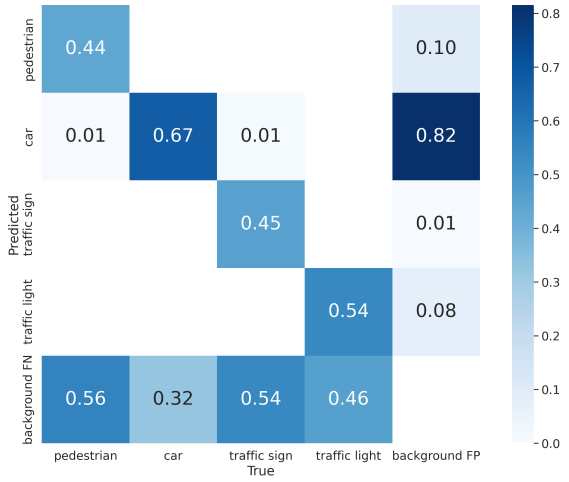


Figure 29. Confusion matrix

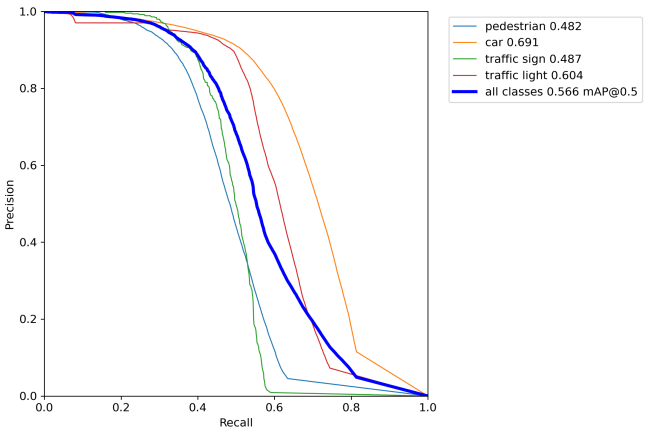


Figure 30. Precision/Recall curve

4.7 Evaluations with real rain

4.7.1 Clean trained YOLOv5 evaluated on Real Rain

After the main evaluations are done, we also want to evaluate with Real Rain and Real Derained datasets, in order to confirm our findings can be useful for real rainy images also. First evaluation we do is for creating a new baseline for the next evaluations, as domain differences exist between Argoverse and BDD100K. mAP for all classes at IoU=0.50 is 0.339.

4.7.2 Synthetic Rain trained YOLOv5 evaluated on Real Rain

With this test we can see that our network trained with occlusions related to rain streaks doesn't perform better with BDD100K's images with mainly rain drops on them. mAP for all classes at IoU=0.50 is 0.311.

4.7.3 Clean trained YOLOv5 evaluated on Derained Real

For this final evaluation, we test to see if Transweather improves the precision we get with images including real rain drops and find out that it isn't the case. mAP for all classes at IoU=0.50 is 0.327.

4.8 Findings

First we evaluated the default setup to use as a baseline, i.e. YOLOv5 trained on the unmodified Argoverse dataset's training subset, and evaluated on the same dataset's validation subset. The resulting mAP score at 0.5 IoU threshold was the highest (0.581) as expected. This is because this is the most optimal setup with no occlusions added whatsoever. This is represented at the 1st row of Table 2.

Then, we evaluated the same network, with the same unmodified training data, but with evaluation set containing the added synthetic rain. This causes the mAP performance at 0.5 IoU to drop 0.071 points to 0.510. This reduced mAP score is also expected, as the model was trained with the training data free from rain-occlusions. So, the network performance degrades when actually tested on images in the validation dataset, which have such occlusions present in them. The result of this experiment is listed in the 2nd row of Table 2.

The third evaluation in the Table 2 represents one of the two pipelines we're comparing in this work: The training data hasn't changed, but for validation data we take the modified Argoverse validation dataset and we run it through the deraining network Transweather [39] to remove the rain from the images before the process of object detection and classification. The mAP score has slightly increased up to 0.519, because of

the deraining process, which improved the images quality by removing the rain effect. But, it's not close to the base-line performance of the 1st experiment.

The fourth evaluation represents the second pipeline we're considering in the scope of this work: We use Argoverse dataset's training subset augmented with rain for the training of the neural network. In this way, we obtain a network that is already aware of the rainy weather conditions. Therefore, we get decent mAP at 0.555, because the network is trained on the same kind of occlusion that appears on the evaluation data. This method also has the benefit of being less resource intensive as it requires running only one neural network, without needing a deraining neural network. The result of this experiment is listed in the 4th row of Table 2.

The fifth evaluation conducted in this work is performed to test whether we lose precision when using a synthetic rainy trained neural network on non-rainy data, as we might expect in real-life situations. The resulting mAP (0.566) shows a minor loss in precision over the base-line mAP.

Real Rain evaluations The next 3 evaluations are conducted for gaining insight on feasibility of the methods on data with real rain in it. The first limitation we face on this one is that the dataset is small in size with only 1596 images, so we'll be using these only as evaluation datasets with training done only with either Clean or Synthetic Rain datasets. Another limitation we face is in regards to domain differences between the Argoverse and BDD100K datasets: They use different kinds of cameras, produce images in different resolutions and the images in Real Rain has various degrees of rain drops on them instead of rain streaks that are present in Synthetic Rain dataset.

We conduct the 6th evaluation on the Table 2 in order to establish a new baseline because of the domain differences. The mAP at 0.339 is lower than evaluations done with the datasets of the same domain as the training dataset, as expected.

The evaluation №7 is conducted to determine if our Synthetic Rain trained network performs better than Clean trained one on images with real rain. It performs worse and this is rather expected, as there's a considerable difference between the rain streaks Synthetic Rain is trained with and the rain drops that exist in Real Rain dataset.

The final and 8th evaluation is comparable to the 6th evaluation, except the evaluation data has gone through Transweather for removing rain drops from the images. While visually the images after this process do look less rainy in comparison, surprisingly the precision of the network actually drops after deraining.

Table 2. The evaluation results

№	YOLOv5 training data	Evaluation data	mAP@0.5
1	Clean	Clean	0.581
2	Clean	Synthetic Rain	0.510
3	Clean	Synthetic Derained	0.519
4	Synthetic Rain	Synthetic Rain	0.555
5	Synthetic Rain	Clean	0.566
6	Clean	Real Rain	0.339
7	Synthetic Rain	Real Rain	0.311
8	Clean	Derained Real	0.327

5 Conclusion

This research provides a useful study for improving object detection in adverse weather conditions for Auve Tech's autonomous vehicle (a shuttle bus). For this purpose, we have used the YOLOv5 model as an object detection model along with a few other tweaks in different experiments with the Argoverse dataset. The images in the Argoverse dataset were augmented with synthetic rain at different levels. Then, different experiments were performed, which included training with the rainy dataset as well as removing the rain effect from images before actually feeding them to the network for inferencing. Table 2 shows that the YOLOv5 network trained with rainy images performed best, in terms of the mAP value (0.555), when compared to the baseline value (i.e. the mAP value when Synthetic Derained is used for evaluation: 0.519). It also performed comparable to the baseline mAP value (0.566), when it was used to test on validation data with no rain. This indicates that the model can be reliably used in real-world situations involving both adverse and normal weather conditions. With the last 3 experiments done we can also see that while deraining provided by Transweather [39] results in more legible images visually, it doesn't improve the object detection performance, actually resulting in slightly lower mAP (down to 0.327 from 0.339 as the baseline value).

Considering the data above and the fact that Transweather would need further optimizations in order to be a viable option to be used in real-time as discussed in Section 3.3.1, we are confident in suggesting Auve Tech an object detection pipeline that includes training YOLOv5 with labeled data in adverse weather conditions.

6 Future work

Firstly, we believe that this work should be revisited in the future when more performant neural networks for removing adverse weather artifacts from images are available as collecting and labeling a large dataset in all kinds of adverse weather conditions is no simple undertaking and might not be viable for Auve Tech.

Recently, Generative Adversarial Networks (GANs) have been achieving very promising performance results in the area of image translation and while they don't exist in a suitable form for Auve Tech's pipeline yet, developments might prove useful in the future as they have the potential to solve the issues we are facing with domain differences.

Looking at the confusion matrices from our experiments, we can see that Background False Positives and False Negatives are quite high. Future work should focus on improving this aspect of the network by including more images without objects in them in order to improve this.

References

[1] T. Nhan, "Global Status Report on Road Safety," tech. rep., World Health Organization, 2018.

[2] M. F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays, "Argoverse: 3D Tracking and Forecasting with Rich Maps," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 8740–8749, 11 2019.

[3] Auve Tech, "Auve Tech Autonomous vehicles and smart transportation systems.," 2019.

[4] R. Sell, R.-M. Soe, R. Wang, and A. Rassõlkin, "Autonomous Vehicle Shuttle in Smart City Testbed," in *Lecture Notes in Mobility*, pp. 143–157, Springer, Cham, 2021.

[5] M. Bellone, A. Ismailogullari, J. Müür, O. Nissin, R. Sell, and R.-M. Soe, "Autonomous Driving in the Real-World: The Weather Challenge in the Sohjoa Baltic Project," in *Towards Connected and Autonomous Vehicle Highways*, pp. 229–255, Springer, Cham, 2021.

[6] D. Wang, S. Jiang, P. Zhang, a. , M. Strayer, G. Tian, J. Zhou, X. Li, M. Malayjerdi, C. Baykara, R. Sell, and E. Malayjerdi, "Safety Assessment and Simulation of Autonomous Vehicle in Urban Environments," *IOP Conference Series: Materials Science and Engineering*, vol. 1140, p. 012032, 5 2021.

[7] K. Kalda, R. Sell, and R.-M. Soe, "Self-driving Shuttle Bus Use Case in City of Tallinn," *IOP Conference Series: Materials Science and Engineering*, vol. 1140, p. 012047, 5 2021.

[8] R. Wang, R. Sell, A. Rassolkin, T. Otto, and E. Malayjerdi, "Intelligent Functions Development on Autonomous Electric Vehicle Platform," *Journal of Machine Engineering*, vol. 20, pp. 114–125, 6 2020.

[9] A. Rassõlkin, R. Sell, and M. Leier, "Development case study of the first estonian self-driving car, iseauto," *Electrical, Control and Communication Engineering*, vol. 14, pp. 81–88, 7 2018.

[10] H. Yaroslav and J.-P. Ernits, "Visuaalne lokaliseerimine Iseautole kasutades ruumilise info tuletamist kaamera liikumisest," Master's thesis, TalTech, 2019.

[11] F. O. Akandere, "Telemetry on robot operating system based self-driving vehicles," Master's thesis, TalTech, 2019.

[12] M.-L. Klaats, “Self-driving car ISEAUTO base frame design,” Master’s thesis, TalTech, 2019.

[13] M. Mirjam Feodorov and J.-P. Ernits, “Dünaamiliselt uuendatav 3D kaardi lahendus Iseautole,” Master’s thesis, TalTech, 2019.

[14] T. Priit, “Autonoomse sõiduki raja järgiminevabavaralise tarkvaraga Autoware,” Master’s thesis, TalTech, 2018.

[15] V. Mihkel, “Evaluation of multiple lidar placement on a self-driving car in Autoware,” Master’s thesis, TalTech, 2018.

[16] L. Roomere and E. Juhan-Peep, “Isejuhtiva auto tarkvara mudelipõhine integratsioonitestimine Autowarenäitel,” Master’s thesis, TalTech, 2018.

[17] A. Vainola, “Estimating object detection reliability for TTU "Iseauto" self-driving car,” Master’s thesis, TalTech, 2018.

[18] K. Aivar, “ISEAUTO Analüüs - Sarnased projektid,” tech. rep., Taltech, 2017.

[19] University of Tartu, “The first autonomous hydrogen vehicle in the world showcased in Tartu | University of Tartu,” 2021.

[20] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii, and T. Azumi, “Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems,” *Proceedings - 9th ACM/IEEE International Conference on Cyber-Physical Systems, ICCPS 2018*, pp. 287–296, 8 2018.

[21] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, “ROS: an open-source Robot Operating System,” *lars.mec.ua.pt*, 2009.

[22] “ROS/Introduction - ROS Wiki.”

[23] “GitHub - tier4/AutowareArchitectureProposal.proj: This is the source code of the feasibility study for Autoware architecture proposal..”

[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 580–587, 9 2014.

[25] R. Girshick, “Fast R-CNN,” *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[26] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 2999–3007, 12 2017.

[27] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, 2016.

[28] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object Detection via Region-based Fully Convolutional Networks," *Advances in Neural Information Processing Systems*, pp. 379–387, 5 2016.

[29] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 779–788, 6 2015.

[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1137–1149, 6 2015.

[31] M. Hnewa and H. Radha, "Object Detection Under Rainy Conditions for Autonomous Vehicles: A Review of State-of-the-Art and Emerging Techniques," *IEEE Signal Processing Magazine*, vol. 38, no. 1, pp. 53–67, 2020.

[32] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv*, 4 2018.

[33] G. Jocher, A. Stoken, A. Chaurasia, J. Borovec, NanoCode012, TaoXie, Y. Kwon, K. Michael, L. Changyu, J. Fang, A. V, Laughing, tkianai, yxNONG, P. Skalski, A. Hogan, J. Nadar, imyhxy, L. Mammana, AlexWang1900, C. Fati, D. Montes, J. Hajek, L. Diaconu, M. T. Minh, Marc, albinxavi, fatih, oleg, and wanghaoyang0106, "ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support," 10 2021.

[34] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 6517–6525, 12 2016.

[35] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 10778–10787, 11 2019.

[36] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common Objects in Context," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5 2014.

[37] K. Garg and S. K. Nayar, "When does a camera see rain?," *Proceedings of the IEEE International Conference on Computer Vision*, vol. II, pp. 1067–1074, 2005.

[38] H. Kurihata, T. Takahashi, I. Ide, Y. Mekada, H. Murase, Y. Tamatsu, and T. Miyahara, "Rainy weather recognition from in-vehicle camera images for driver assistance," *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2005, pp. 205–210, 2005.

[39] J. M. J. Valanarasu, R. Yasarla, and V. M. Patel, "TransWeather: Transformer-based Restoration of Images Degraded by Adverse Weather Conditions," in *CVPR*, 11 2022.

[40] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[41] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," *CVPR*, 4 2016.

[42] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 5 2018.

[43] A. B. Jung, "imgaug," 2020.

[44] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, 4 2004.

Appendix

I. Glossary

- Artificial Neural Network: A biologically inspired computational model used for complex function approximation tasks.
- Convolutional Neural Network: A variation of conventional Artificial Neural Network mainly used for working with image data as convolutional layers help reduce parameters of the network.
- Object detection: A computer vision task for detecting (localizing) objects of interest in an image.
- Autonomous Vehicles: A driverless car, which is able to navigate without human intervention.
- Shuttle: a vehicle used to transport passengers between 2 (or more) frequent places.
- Computer vision pipeline: A set of computer vision based methods applied in sequential steps, in which the subsequent step is dependent on the previous step(s).
- Synthetic dataset: A dataset that is artificially generated using computational (or other artificial) methods.
- Restoration: The process of restoring images to their original form improve its quality.
- Smart transportation: The use of modern technology for building an integrated transportation system.
- autonomous transportation: The use of modern technology for building an intelligent transportation system.
- LIDAR: Stands for Light Detection and Ranging; a remote-sensing method for measuring distances in the form of point-cloud data.
- Pointcloud data: A set of points in 3D space for representing a 3D object or world.
- GPU: Stands for Graphical Processing Unit; a processor dedicated to intensive computations in particular.
- Robot Operating System: An open-source meta-operating system for robots.

- Bounding box: A set of 4-values representing the x-y coordinates of extremes of a rectangle around an object of interest.
- Object tracking: The process of detecting an object and then following it by capturing its location at any moment in time.
- Multi object tracking: Tracking multiple objects simultaneously using computer vision based methods.
- Sensor fusion: Fusing the data of multiple sensors to produce a more higher dimensional representation.
- Region Proposal Network: A specialized network for generating proposals (potential ROIs) for objects of interest.
- Region of Interest: The area in image where an object of interest might be located.
- YOLO: Stands for 'You only look once'; A breakthrough computer vision single-stage object detection model for real-time object detection.
- R-CNN, Fast R-CNN, Faster R-CNN: Different variations of 2-stage object detection CNN models.
- Intersection over Union: The ratio of area of overlap with the combined area (union) of 2 bounding boxes.
- YOLOv3, YOLOv5: More accurate and efficient variations of the original YOLO model by the same author.
- DarkNet: An efficient open-source Convolutional Neural Network written in C language.
- Deraining: The process of removing rain pixels from an image.
- Desnowing: The process of removing snow pixels from an image.
- Defogging: The process of removing fog pixels from an image.
- Transweather: An efficient network for deraining.
- Confusion matrix: An NxN square matrix used to calculate different classification measures, where N-1 is the number of classes.
- precision: The ratio of correct positive prediction to total positives.
- recall: The ratio positives correctly identified as positives.

- Training dataset: The set of examples set out for training a machine learning model.
- Validation dataset: The set of unseen examples set out for validating an already trained machine learning model
- Argoverse: A dataset designed for support the autonomous vehicles research.

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Elmar Abbasov**,

1. grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright, my thesis

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Improving object detection in adverse weather conditions for Auve Techs autonomous vehicle,

   supervised by Gholamreza Anbarjafari and Asif Sattar.

2. I grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in points 1 and 2.

4. I confirm that granting the non-exclusive licence does not infringe other persons intellectual property rights or rights arising from the personal data protection legislation.

Elmar Abbasov
24.05.2022